SIT232 Object Oriented Development
Task 1.3D

By Venujan Malaiyandi
BSCP|CS|61|101

Comprehensive Tutorial: Network Utility Program in C# with Object-Oriented Development

Greetings! I am Venujan Malaiyandi, and I'm delighted to present this tutorial on network utilities, a fundamental aspect of modern computing. As part of my assignment for the Object Oriented Development Module, Task 1.3D - Helping Others, I aim to provide you with a comprehensive guide to building network utilities using the power of Object Oriented Development in C#. In this tutorial, we will delve into the world of network programming, exploring how object-oriented principles can be leveraged to create robust and efficient solutions for tasks ranging from data transmission to network analysis.

Introduction

Welcome to this comprehensive tutorial on creating a Network Utility program in C# using Object-Oriented Development! This tutorial will guide you through building a console-based application that leverages object-oriented principles for tasks like resolving domains to IP addresses, pinging addresses, and performing nslookups.

Prerequisites

Before diving into this tutorial, it's recommended to have:

- Basic understanding of C# programming.
- Familiarity with object-oriented programming (OOP) concepts (classes, objects, properties, methods, etc.).

- Knowledge of networking basics (IP addresses, domain names, ping, etc.).
- An integrated development environment (IDE) for C# (e.g., Visual Studio).

Object-Oriented Development Concepts

Let's connect the concepts of Object-Oriented Development with the components of our program:

1. PingService Class

Purpose: Encapsulates the functionality to send ping requests.

Object-Oriented Concept:
Encapsulation: It bundles related data (properties) and behavior (methods) into a single unit.

2. DnsResolver Class

Purpose: Resolves a domain name to its corresponding IP address.

Object-Oriented Concept:
Static Method: It provides a utility function (ResolveToIP) that can be called without instantiating an object. This is useful for operations that don't require maintaining state.

3. NslookupService Class

Purpose: Performs an nslookup operation on a given domain.

Object-Oriented Concept:
Abstraction: It hides the underlying complexity of using the `nslookup` command and provides a simple method (`PerformNslookup`) for users.

4. Program.cs

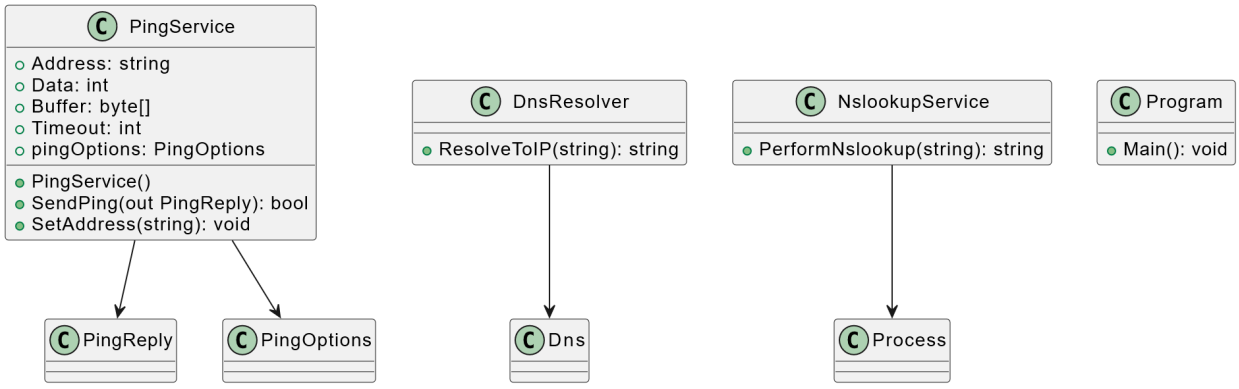
Purpose: Contains the entry point of the application and handles user interaction.

Object-Oriented Concept:

- **Instantiation** - It creates an instance of the `PingService` class (`pingService`) to utilize its functionality.
- **Control Structures:** It uses a `while` loop and a `switch` statement to control the flow of the program.

UML diagram:

The UML diagram for this tutorial is attached as an svg. Additionally, it's pasted as the image below.



Step-by-Step Guide

Step 1: Creating PingService.cs

Class Definition:

- Define the PingService class within the NetwkUtility namespace.
- Declare properties (Address, Data, Buffer, Timeout, pingOptions) to store ping-related information.
- Implement a constructor to set default values and initialize pingOptions.

Method to Send Ping:

- Create a method (SendPing) that uses the Ping class to send a ping.
- Return whether the ping was successful.

Dynamic Address Setting:
Implement a method (SetAddress) to allow dynamic setting of the address.

Step 2: Implementing DnsResolver.cs

Class Definition:
Define the DnsResolver class within the NetwkUtility namespace.

Static Method for DNS Resolution:

- Create a static method (ResolveToIP) that takes a domain name as input.
- Use Dns.GetHostAddresses to retrieve IP addresses and return the first address as a string.

Step 3: Building NslookupService.cs

Class Definition:
Define the NslookupService class within the NetwkUtility namespace.

Performing nslookup Operation:

- Create a static method (PerformNslookup) that takes a domain name as input.
- Use Process to execute the `nslookup` command and capture the output.

Step 4: Crafting the Program.cs

Main Method:
Define the `Main` method as the entry point of the application.

Menu-Driven User Interaction:
Implement a menu-driven loop that presents options to the user (resolve domain, ping, nslookup, exit).

Utilize Utility Classes:

- Instantiate`PingService (pingService) to utilize its functionalities.

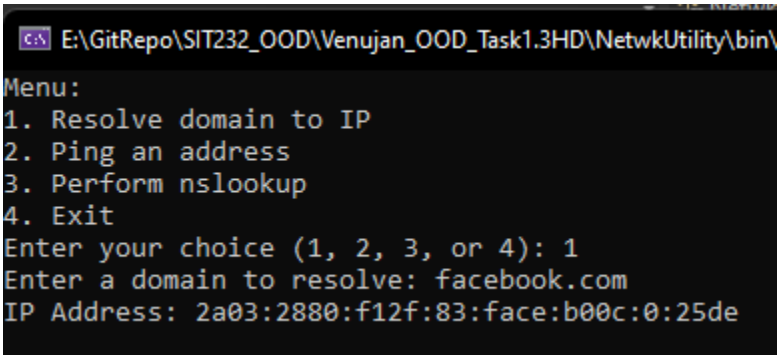
- Based on user input, interact with the respective utility classes (DnsResolver, NslookupService) to perform tasks.

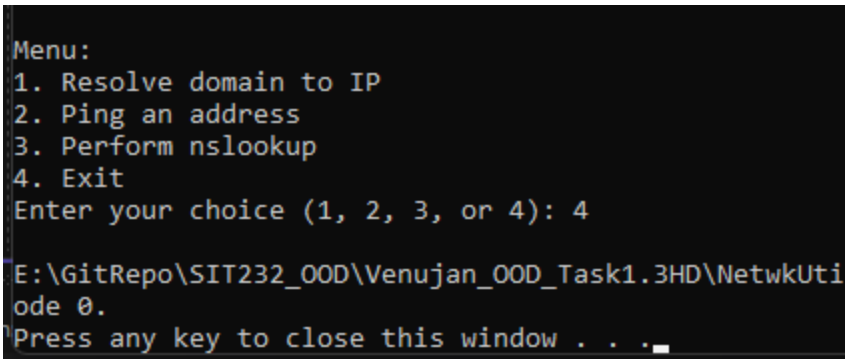
Step 5: Running the Application

Compile and run the application. Follow the prompts for each operation to see the results.

Output

The compiled output for the program is given below. You may use them as reference to debug your application.





Conclusion

You have successfully built a Network Utility program in C# using Object-Oriented Development principles! This program demonstrates encapsulation, abstraction, static methods, and control structures in action. Feel free to expand and enhance the program further, and explore other OOP concepts like inheritance and polymorphism for even more advanced functionalities.
Happy coding!